Auteur : Spinelli Isaia

Prof : [Etienne](https://cyberlearn.hes-so.ch/user/view.php?id=104149&course=1) Messerli

Ing : Sébastien Masle

Date : 22.10.2019

Salle : A09 – HEIG-VD

Classe : CSN

Additionneurs en VHDL

[CONCEPTION DE SYSTÈMES NUMÉRIQUES (CSN)](https://cyberlearn.hes-so.ch/course/view.php?id=14116#section-1)

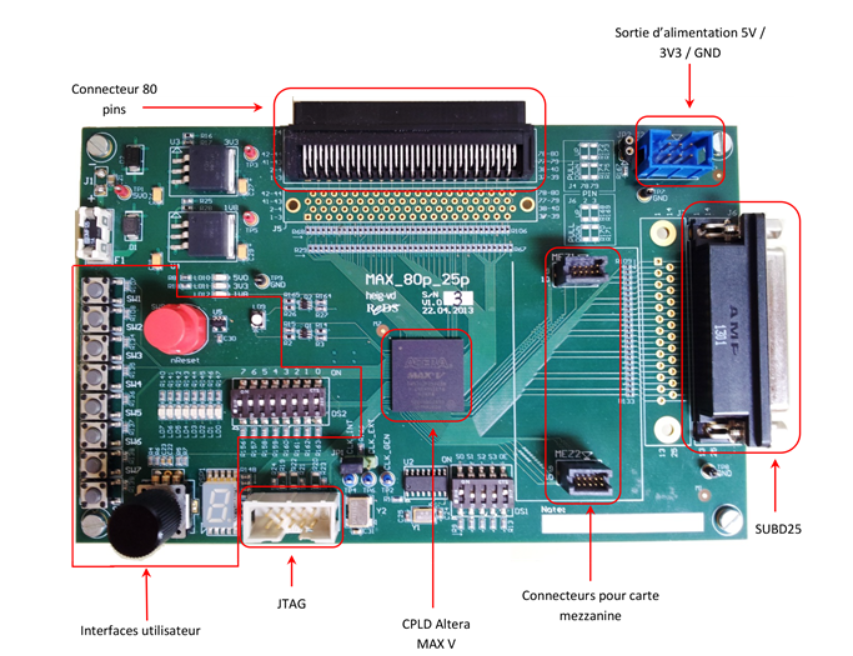


Table des matières

[Introduction - 2 -](#_Toc22115842)

[Encodeur de priorité à 4 entrées - 2 -](#_Toc22115843)

[Spécifications - 2 -](#_Toc22115844)

[Description textuelle - 3 -](#_Toc22115845)

[Description en VHDL - 4 -](#_Toc22115846)

[Simulation - 5 -](#_Toc22115847)

[Synthèse - 7 -](#_Toc22115848)

[Encodeur de priorité à 16 entrées - 8 -](#_Toc22115849)

[Analyse - 8 -](#_Toc22115850)

[Description en VHDL - 10 -](#_Toc22115851)

[Simulation automatique - 10 -](#_Toc22115852)

[Synthèse - 10 -](#_Toc22115853)

[Intégration - 11 -](#_Toc22115854)

[Test de l’intégration - 11 -](#_Toc22115855)

[Conclusion - 12 -](#_Toc22115856)

[Difficultés rencontrées - 12 -](#_Toc22115857)

[Compétences acquises - 12 -](#_Toc22115858)

[Résultats obtenus - 12 -](#_Toc22115859)

## Introduction

Description de différentes versions de l'addition en VHDL par étapes, soit :

* Sans dépassement
* Avec carry
* Avec carry et overflow

## Additionneur 4 bits avec carry (in et out)

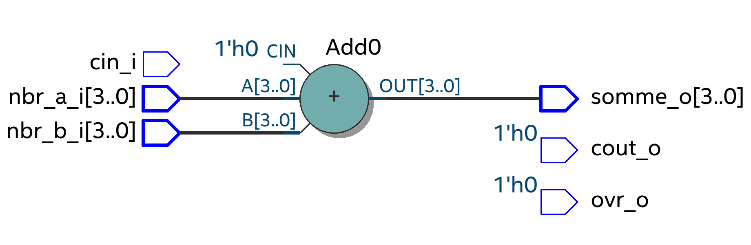
Le code est en annexe.

Suivis des vues RTL :

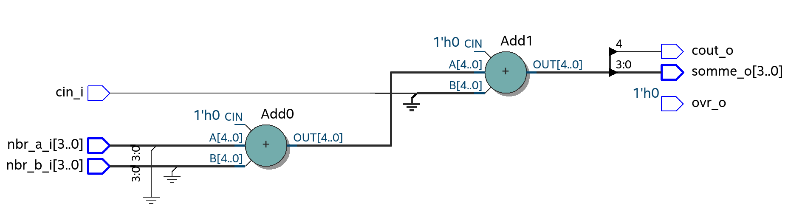
Vue RTL étape a : additionneur de deux vecteurs 4 bits sans report



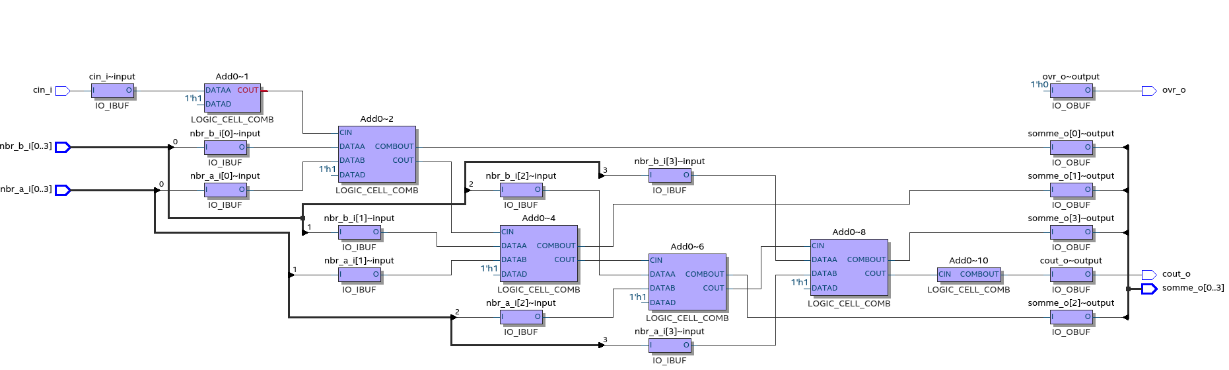
Vue RTL étape b : ajout du carry out



Vue RTL étape c : ajout du carry in



Vue Technologique de l’étape c :



Remarque : On peut voir dans la vue RTL pour l’ajout du carry in, qu’il a besoin d’ajouter un additionneur pour utiliser le CIN mais dans la vue Technologique on constate qu’il utilise seulement 4 additionneurs 1 bits.

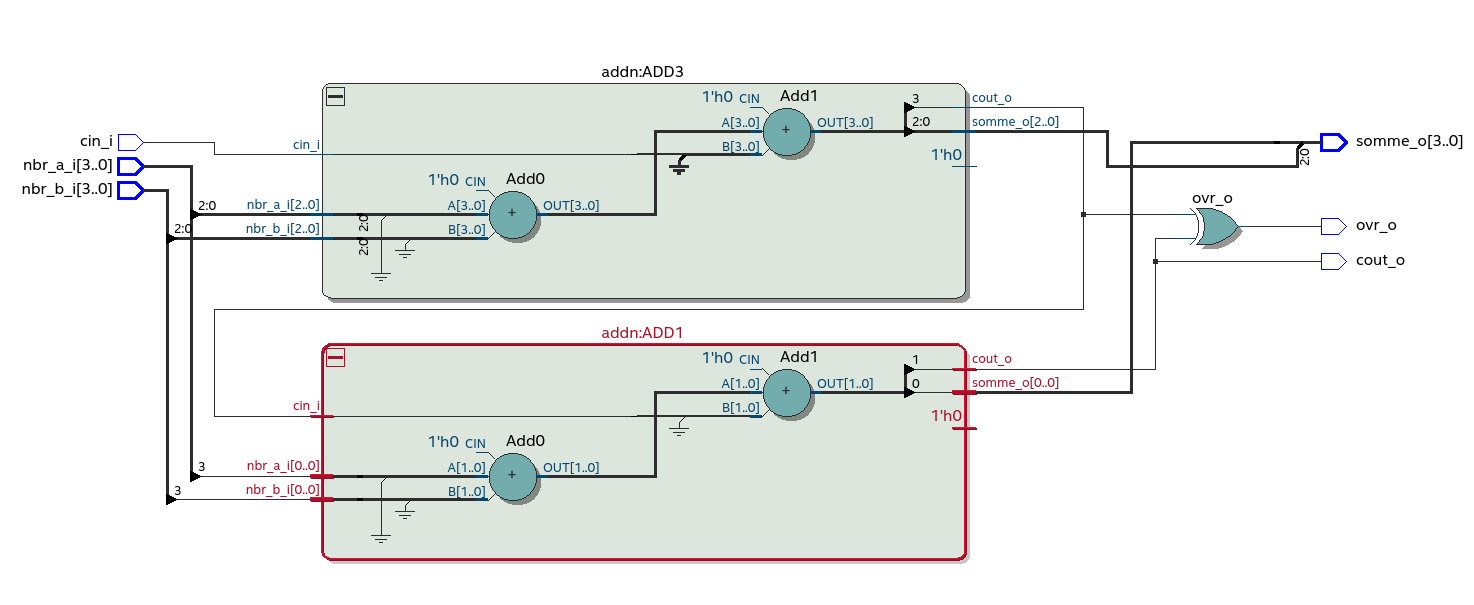
## Additionneur générique N bits avec carry (in et out)

Le code est en annexe.

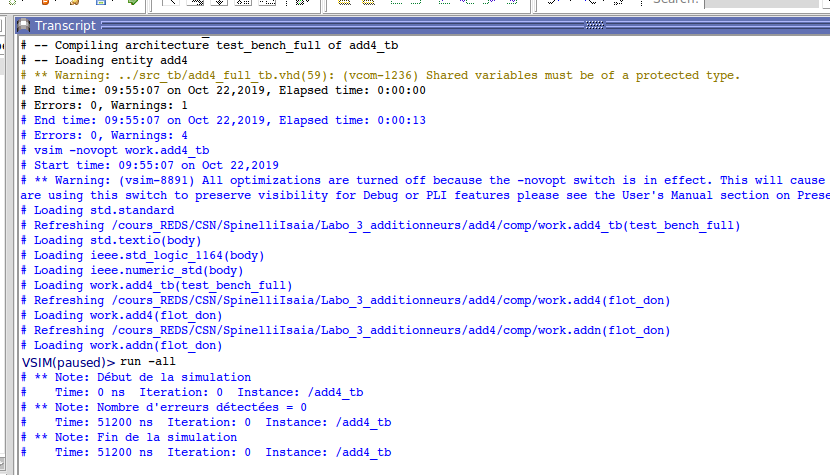
## Additionneur 4 bits avec carry (in et out) et overflow

Le code est en annexe.

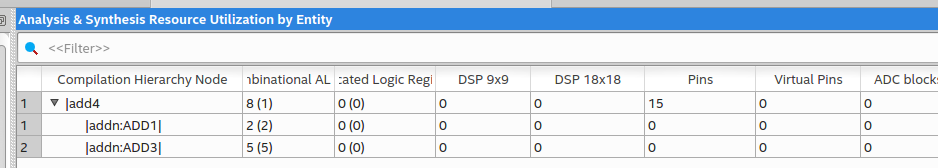
Voici la vue Technologique qui donne une bonne vue d’ensemble de la logique de l’additionneur avec overflow :



Voici le log de la simulation automatique de l’add4\_full :



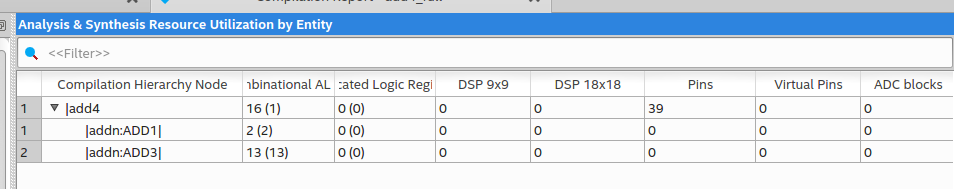
Voici la logique avec l’additionneur complet 4 bits :



## Additionneur générique N bits (in et out) et overflow

Le code est en annexe.

Voici la quantité logique de l’additionneur générique avec overflow pour (12 bits)



Remarque : On peut voir que l’additionneur ADD1 a bien évidemment la même quantité de logique et que le ADD3 augmente de façon linéaire N+1 quantité logique N étant le nombre de bit.

Date : 22.10.19

Nom de l’étudiant : Spinelli Isaia